CS 590 Final

This is a closed book, closed notes examination, with the following exception: you may refer to a single 8.5x11 inch (or A4) sheet of notes—with anything on it which you think might be useful. You may also use as much blank scratch paper as you need.

This exam is worth 100 points, and each problem is worth 10 points.

It is a two hour exam.

Note: answers with proofs, or at least plausible arguments, will receive more credit than those without.

CS 590 Theory of Computation. Test T

May 1993

Problem T-1:

Let $Fin = \{e: W_e \text{ is finite}\}$. Is $Fin$ recursive?

Answer:

$Fin$ is a non-trivial index set, so it is not recursive by Rice’s theorem.

Problem T-2:

Construct a set which is undecidable relative to $K$—that is, it is undecidable even when $K$ is used as an oracle. (Hint: diagonalize).

Answer:

Let $W_e(K)$ be the set you can recognize with Turing machine $M_e$ and oracle $K$. Let $K(K) = \{e: e \notin W_e(K)\}$. $K(K)$ cannot be recognized by any machine $M_e$ even with the oracle, since it differs from every such set on the value $e$.

Problem T-3:

Prove that there is a total, recursive function $m$ such that

$$E_{m(x)} = \{0, 1, 2^x, 3^x, \ldots\}$$

Answer:

Let $m$ be such that $\phi_{m(x)}(y) = y^x$.

There is an $f$ as required.
Problem T-4:
Prove that if $LOG = NLOG$ then $DSPACE(n) = NSPACE(n)$.

Answer:
A simple padding argument.

Problem T-5:
Prove that there is an oracle $A$ such that $P(A) = PH(A)$. (Hint: $PH(A) \subseteq PSPACE(A)$ for all $A$).

Answer:
Let $A = QBF$. Then $P(A) \subseteq PH(A) \subseteq PSPACE(A) = PSPACE \subseteq P(A)$

The first relation follows from the fact that $\sigma_0^p(A) = P(A) \subseteq \sigma_k^p(A) \subseteq PH(A)$ for any oracle $A$—which follows from the fact that the proof of Wrathall’s theorem relativizes. Similarly, $PH(A) \subseteq PSPACE(A)$ for any oracle, since the proof that $PH \subseteq PSPACE$ relativizes. We know that $PSPACE(QBF) = PSPACE$ as we saw in class. The last relation follows from the fact that every set in $PSPACE$ is decidable in polynomial time with access to $QBF$, since $QBF$ is $PSPACE$-complete.

Problem T-6:
Suppose the function $f$ is computable using polynomial time (i.e., $f \in PF$). How complex is the set $G_f = \{y : f(x) = y\}$ for some $x$?

Answer:
It is in $NP$ by Wrathall’s theorem, since

$$G_f = \{y : \exists^p x (f(x) = y)\}$$

and the predicate in parentheses is polynomial time decidable. Note that $f(x)$ is at most polynomial longer than $x$, since only polynomial time is available to compute $f$. That is, for some polynomial $p$, $|x| \leq p(|f(x)|)$.

Problem T-7:
Prove that $\leq_m^p$ is reflexive and transitive (i.e., for any $A$, $B$, and $C$, $A \leq_m^p A$ and if $A \leq_m^p B$ and $B \leq_m^p C$ then $A \leq_m^p C$).

Answer:
See the book.
Problem T-8:
Say a nondeterministic Turing machine accepts a language unambiguously if it has exactly one accepting path in its computation tree for every string in the language (and none for those strings not in the language). Let $UP = \{ L : \text{some NTM } M \text{ accepts } L \text{ unambiguously in polynomial time} \}$. What is the relationship between $P$, $NP$, and $UP$?

Answer:
$P \subseteq UP \subseteq NP$.
Any DTM running in polynomial time is also an NTM in poly time with exactly one path. So $P \subseteq UP$.
Also, if $L \in UP$ then there is a poly time NTM which accepts it, so $UP \subseteq NP$.

Problem T-9:
Prove that $P(SAT) = NP$.

Answer:
Let $L \in P(SAT)$ via OTM $M$. Replace every query to $SAT$ with a call to a nondeterministic polynomial time algorithm for $SAT$. The new algorithm is poly time (composition of the poly for $M$ and that for the $SAT$ NTM). And it’s non-deterministic. So $L \in NP$.
Conversely, let $L \in NP$. $f : L \leq_{m} P SAT$ for some poly time computable $f$, since $SAT$ is $NP$ complete. For input $x$, compute $f(x)$ and ask the oracle if $f(x) \in SAT$. This is a poly time algorithm which queries $SAT$, so $L \in P(SAT)$.

Problem T-10:
What is wrong with the following argument:
Let $A = \{ x = 1^i : x \notin L(M_i) \}$ where $M_0, M_1, \ldots$ lists all polynomial time DTMs. Clearly, by diagonalization, $A \not\in P$.
To recognize $A$, use the following non-deterministic algorithm:

Input $x$
If $x = 1^n$
Then
  Guess an accepting computation of $M_n$ on input $x$
  Verify that this is indeed a correct guess
Reject
Else reject
Accept

This algorithm is clearly polynomial time, since each simulation is polynomial time. It is correct because it can guess an accepting path only if one exists, in which case \( x \notin A \). On the other hand, if no accepting path exists, in which case \( x \in A \), then this algorithm accepts. So, this is an \( NP \) algorithm for \( A \).

So, \( A \in NP - P \) and \( NP \neq P \).

Answer:
First, the enumeration of DTMs may not be effective in poly time, so verifying the guess may not be poly time since just getting a description of \( M_n \) from \( n \) may take too long.

Second, even if a good enumeration were chosen, the length of the computation path is the polynomial bounding \( M_n \), and this will differ from machine to machine. So, we cannot conclude that there is a single polynomial which bounds the runtime of this algorithm.